2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Domain Transfer for Semantic Segmentation of LiDAR Data using Deep Neural Networks

Ferdinand Langer Andres Milioto Alexandre Haag Jens Behley Cyrill Stachniss

Abstract— **Inferring semantic information towards an understanding of the surrounding environment is crucial for autonomous vehicles to drive safely. Deep learning-based segmentation methods can infer semantic information directly from laser range data, even in the absence of other sensor modalities such as cameras. In this paper, we address improving the generalization capabilities of such deep learning models to range data that was captured using a different sensor and in situations where no labeled data is available for the new sensor setup. Our approach assists the domain transfer of a LiDAR-only semantic segmentation model to a different sensor and environment exploiting existing geometric mapping systems. To this end, we fuse sequential scans in the source dataset into a dense mesh and render semi-synthetic scans that match those of the target sensor setup. Unlike simulation, this approach provides a real-to-real transfer of geometric information and delivers additionally more accurate remission information. We implemented and thoroughly tested our approach by transferring semantic scans between two different real-world datasets with different sensor setups. Our experiments show that we can improve the segmentation performance substantially with zero manual re-labeling. This approach solves the number one feature request since we released our semantic segmentation library LiDAR-bonnetal [18].**

I. Introduction

In autonomous driving, a precise, reliable, and fast understanding of the scene is crucial for operation. Most self-driving cars use onboard sensors to perceive the scene around the vehicle and semantic segmentation is an important subtask of scene understanding. Semantic segmentation assigns a class label to each data point, i.e., a pixel or a 3D point, and allows autonomous vehicles to perform a wide variety of tasks, such as obstacle avoidance, tracking other participants, traversability analysis, and many more. To perform semantic segmentation accurately and robustly in various scene conditions, multiple sensors, like cameras, radars, and LiDARs, are used to achieve the needed redundancy, as well as coverage of the whole scene or recording conditions. While it is typically advantageous to rely on multiple sensor modalities, there are conditions in which a vehicle must rely solely on active sensors like LiDAR, e.g., when driving at night. In this work, we focus on the task of LiDAR-only semantic segmentation and specifically on the domain adaption among different sensor configurations.

Many LiDAR datasets [1], [3], [8], [9] are publicly available but only a few of them [1], [9] provide semantic segmentation labels due to the required labor-intensive annotation work. Most of the computationally efficient approaches to semantic segmentation of LiDAR scans rely on a spherical projection of point clouds, see Fig. 2 for an illustration. These approaches exploit the sensor arrangement of most 3D LiDAR sensors, which often consists of a rotating array of lasers. By transforming the scans into a 2D image representation, we can resort to well-studied 2D convolutional neural networks (CNN) for the segmentation task. In addition to that, we can find neighbors in the point clouds requiring less computational resources compared to approaches relying on unordered point clouds. Both properties together allow for building real-time capable segmentation approaches [18], [35]. These approaches perform well in practice, but the projection model uses the specific parameter of the employed sensor, i.e., the position on the car, angular resolution, and field of view (FOV). Applying such a model to LiDAR data recorded in a different domain due to a change in any of these parameters degrades its performance. Although this problem is more severe with projection-based approaches, such degradation also happens with approaches using raw point clouds, since a change in the sensor location or beam distribution changes the overall appearance of the clouds.

The main contribution of this paper is a sensor-oriented transfer method that allows us to exploit existing labels

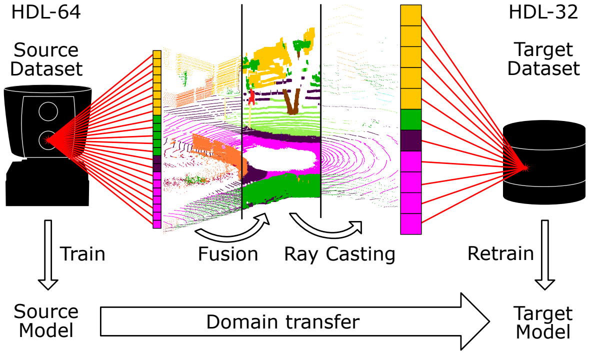


Fig. 1: Proposed domain transfer method that readjusts a semantic segmentation network model to the domain of a different LiDAR sensor.

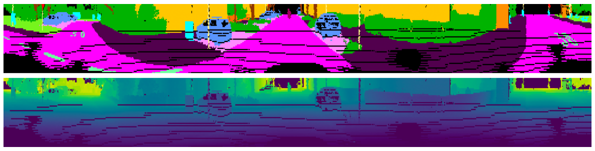


Fig. 2: Sample scan in the projection-based representation (bottom) and its semantic segmentation (top).

F. Langer, A. Milioto, J. Behley, and C. Stachniss are with the University of Bonn, Germany. A. Haag is with Argo.ai, Munich, Germany. This work has partly been supported by the German Research Foundation under Germany's Excellence Strategy, EXC-2070 - 390732324 (PhenoRob) as well as BE 5996/1-1, and by the NVIDIA Corporation.

978-1-7281-6211-9/20/$31.00 ©2020 IEEE

provided for a specific sensor setup and use it in a new setup. One can see this approach as a calibration procedure for a new sensor setup. It runs offline translating the labeled data after the change in the platform configuration. Our approach runs in a couple of hours due to our GPU-enabled rendering pipeline and it generates a model that runs as fast as the original one after our offline adaption. We achieve this by aggregating sequential scans from a semantically annotated dataset such as SemanticKITTI [1] into a dense mesh and sampling realistic 3D LiDAR scans for the new sensor configuration. This allows us to transfer a model trained with the source scanner to a new target scanner. Fig. 1 depicts an example where the target sensor features a lower resolution and a larger vertical FOV than the source sensor.

In sum, we make three key claims: Our approach is able to (i) generate real-to-real LiDAR sweep transfer, including remission information, (ii) transfer the dense semantic annotations to the new sensor setup, and (iii) reduce the domain shift, when adapting to a different LiDAR sensor.

II. Related Work

Point cloud semantic segmentation provides point-wise labels for the whole scene. Approaches for semantic segmentation can be mainly categorized in point-based approaches operating directly on the (sub-sampled) three-dimensional points [23], [24], [32], [15], [30], [11] and projection-based approaches operating on a different representation, like two-dimensional images [18], [35], [38] or three-dimensional regular subdivisions [31], [28], [25].

SqueezeSegV2 [35] performs semantic segmentation of selected road-objects from 3D LiDAR point clouds. In addition to using a real LiDAR dataset for training, they refine the model with synthetic range images generated from a game engine. They use a CNN to predict the remissions not simulated by the game engine. Furthermore, they use domain adaption during training as well as a post-processing step to improve their results. In contrast, we generate realistic-looking scans from a real LiDAR dataset and also interpolate the remissions from real data. We also use all classes provided by SemanticKITTI [1] and are not restricted to their subset of classes, i.e., car, pedestrian, and cyclist. We use for our study RangeNet++ [18], which uses a similar strategy as SqueezeSegV2, but uses a larger backbone and introduced k-nearest neighbor search on the input point cloud they can output fine-grain semantics without "shadowed" artifacts from back-projecting into 3D space.

While aforementioned projection-based approaches [18], [35] to spherical images particularly tend to suffer from a change in the LiDAR sensor setup, like a change in the mounting of the sensor affecting the field-of-view, all other approaches are similarly affected when changing the LiDAR sensor geometry completely, like reducing the number of beams from 64 to 32, which affects the density and pattern of the beam on the surfaces.

Semantic mapping aggregates semantic information into a map representation, which then can be used for other tasks. Approaches generating dense representations, such as truncated signed distance function (TSDF) [34], [36], voxels grids [7], or surfels [4], could be leveraged to simulate a LiDAR scan in the target domain via ray casting. In contrast to these approaches, we target the usage of annotated point clouds instead of segmentation results from images [34], [36] or LiDAR [4] and employ geodesic correlation alignment to reduce the domain shift. We furthermore investigate how the aggregation of point clouds compares to denser representations like a TSDFs for generation of synthetic point clouds.

Domain adaption is a subdomain of transfer learning that assumes the same task but within a different domain. Deep neural networks are tuned for a specific task and trained with a distinct domain defined by the training data. When applying a model to a dataset from a different domain the performance usually drops [21] due to the domain shift. Domain adaptation aims to bridge this gap between source and target domain and to mitigate the degradation of the performance of the network. Some approaches try to avoid retraining by combining multiple sensing modalities [27], [33]. In contrast to that, the aim of our paper is adapting the weights of a deep neural network to another dataset without utilizing human labeling for the target dataset.

Domain adaptation is often needed to improve real-world model performance using simulated data. The simulated scans and the pixel-wise ground truth are sometimes generated using game engines [14], [35] or dedicated simulators [5]. To refine simulated images to look more realistic Generative Adversarial Networks (GAN) are often used [10], [26]. Fernando *et al.* [6], instead try to align the features of CNN directly, rather than generating matching data or augmenting the data to improve realism. To this end, they align the subspace defined by a number of largest eigenvalues through a Principal Component Analysis of both source and target data. This way, a transformation matrix is learned that maps features from the source to the target subspace.

Other approaches suggest introducing a loss that penalizes domain differences during training [19], [29], [35]. These methods can be added to already existing networks, which learn the task with labels from the source domain, but keep the distributions of the activations for both source and target similar. Based on these types of approaches, we employ a domain loss to minimize domain shift by aligning second-order statistics of activations. Unlike these approaches, where the ground truth is rendered from simulated data, our approach renders the scans and labels from meshes that were generated from real data. In contrast, our method allows us to use real remissions and the noise distribution of the source sensor.

Concurrently or after submission of our work, Jaritz *et al.* [12] proposed a framework to perform cross-modal domain adaptation for 3d semantic segmentation using images, Jiang *et al.* [13] propose to use GAN-based domain adaption to transfer labels from a source to a target domain, and Yi *et al.* [37] propose to first create a dense representation from sparse point clouds via a scene completion approach to then learn a classifier on this so-called canonical domain, which can then be used on the canonical domain representation of the target domain.

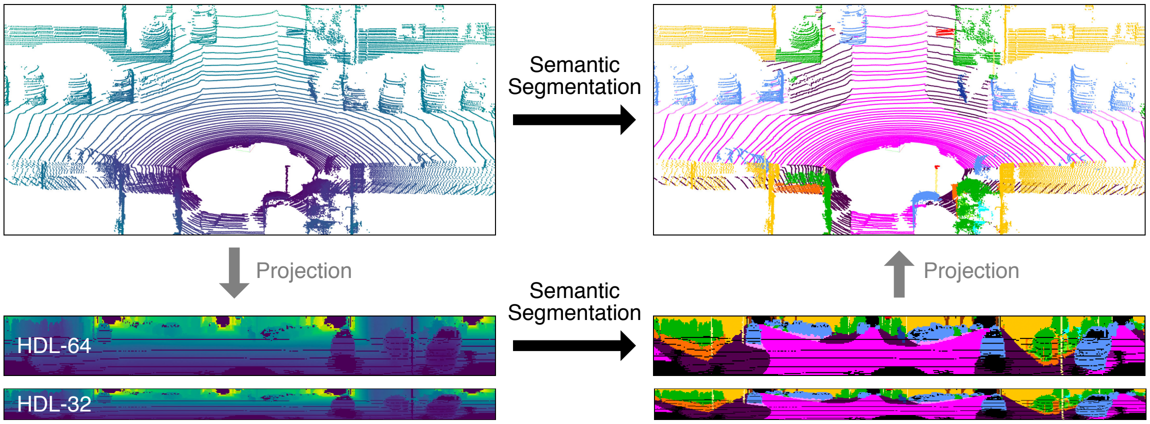


Fig. 3: Directly using point clouds for semantic segmentation vs. employing a projection-based method and projecting the semantics back into the point cloud. Point cloud and range image colored on the left by distance and on the right by class.

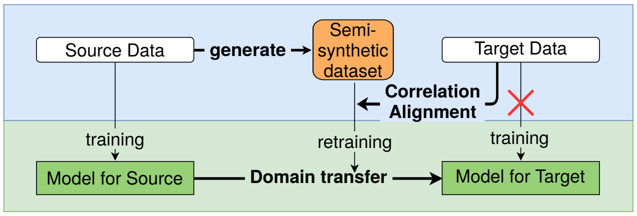


Fig. 4: We achieve the domain transfer by retraining with a semi-synthetic dataset and Correlation Alignment.

III. Our Approach

The goal of our domain transfer approach is to adjust a semantic segmentation network to perform well in the domain of a different LiDAR sensor and/or LiDAR configuration. In our example, we aim to transfer Velodyne HDL-64 scans to match scans from a Velodyne HDL-32, a sensor with a lower resolution and different FOV, as illustrated in Fig. 3.

The key steps of our approach are the following. First, we use annotated scans and build a map of the environment using a 3D LiDAR SLAM approach. From this annotated point cloud, we render realistically looking, semi-synthetic scans. These scan looks as if they would have been recorded from a LiDAR with different sensor parameters. Fig. 4 illustrates the simulation of the semi-synthetic scans.

The second step aims at adapting the segmentation CNN for projected scans to the target domain. We use our generated semi-synthetic dataset to retrain the CNN model that had been originally trained on the source dataset. We use an unsupervised method to align the distributions of the semi-synthetic and real dataset. During the retraining, we calculate an additional loss that aligns the different distributions. Note that our approach uses only raw point clouds from a target sensor and does not require any new labels.

A. Semi-synthetic Scan Simulation

This section describes the generation of semi-synthetic scans. They are built *from* the source domain data and thus come with semantic annotations but are generated *for* the target domain sensor configuration.

The first step of the fusion is to estimate the position of each laser endpoint in a global reference frame using a 3D SLAM pipeline, for which we use SuMa [2]. This allows us to aggregate the point clouds using the SLAM output into a 3D model. This model can either be a large point cloud or also a mesh. The generated 3D representation is then used to simulate the semi-synthetic observations, which look as if they have been obtained by the target sensor, including the corresponding label for each point.

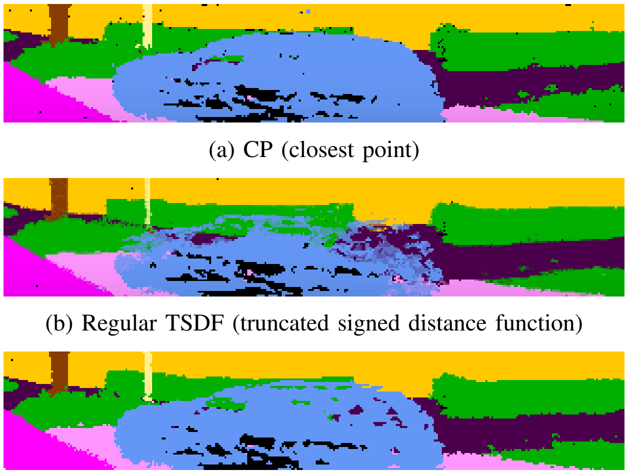
We aim at computing for each scan in the source dataset a scan in the target domain separately. We refer to the primary scan as a single 3D scan from the source dataset for which the translation should be made. We take the pose of this primary scan and render the target semi-synthetic scan for that pose. As our scanner for the target domain might have a higher resolution than in the source domain, a different FOV, or other extrinsic parameters, we consider multiple scans taken before and after the primary scan for the generation of the semi-synthetic scan.

In addition to that, we remove the points that are classified in the ground truth as “moving objects” from all scans but the primary scan to ensure that we only add points from static road objects in that case. Otherwise, multiple instances of the moving objects at different locations would appear in the target scan multiple times leading to wrong observations.

Closest Point (CP). The CP method selects the closest points within the large source point cloud, merged from multiple scans, to build a scan for the target domain. This is achieved by selecting for every pixel of the range image for the target scan the best fitting point from the cloud. First, the merged cloud is projected into a range image to create a simulated scan. This simulated scan with the desired height H and width W by applying a spherical projection, to each 3D point $P = (x, y, z)$:

$$\begin{pmatrix}\psi \\ \phi\end{pmatrix} = \begin{pmatrix}\frac{1}{2}\left[1 - \arctan(y, x)\pi^{-1}\right] W \\ \left[1 - \left(\arcsin(z\ d^{-1}) + f_{up}\right) f^{-1}\right] H\end{pmatrix}, \qquad (1)$$

where $d = ||P||_2$ refers to the distance of P from the origin and $f = |f_{up}| + |f_{down}|$ refers to the vertical FOV of



(a) CP (closest point)

(b) Regular TSDF (truncated signed distance function)

(c) MESH-C (Class-aware TSDF)

Fig. 5: Comparison of generated range images by CP, regular TSDF and MESH-C when integrating 10 scans.

the LiDAR scanner. Second, due to ambiguous assignments of points to the same pixel (ψ, ϕ) in the range image, we select the point with the smallest measured distance, as is usually done in rendering pipelines employing a so-called z-buffer [22]. Other methods like selecting the point minimizing its projection error, i.e., the remainder of the resulting x and y coordinates, would lead to errors due to occlusions.

Mesh. We also use a volumetric method for fusing the point clouds which is widely used in robotics, known as the truncated signed distance function or TSDF [20]. Integrating distances into the TSDF volume for fusion works well for the distance information and the remissions, which are manufacturer-defined reflectance values, serving as the texture of the surface. However, the integration of class labels is not straightforward.

The regular update rule for incrementally building a TSDF from range images integrates the observations from different scans and outputs a weighted sum of the individual observations. In the case of the class labels, this is not desired, since they are represented by a discrete set. Another issue is as the perspective shifts by the movement of the sensor, where we aggregate points from views that are occluded in the primary scan. This means that if a portion of a foreground object is missing, such as the car in Fig. 5 (b), we are at risk of rendering into our image views that are not feasible in reality, such as seeing the grass in the background through the missing roof of the car. To solve this, we propose the following three methods to fuse observations into the TSDF volume (also depicted in Fig. 6):

MESH-A (MA) uses an all-in method that merges the point clouds first, by applying the pose transformation and appending to a joint point cloud. This merged point cloud is then used to generate a single range image, which is integrated to the TSDF. During the projection into the range image, a lot of the points are rejected as there can only be a single link from a 3D point to its 2D counterpart, favoring closer points. Therefore, it favors points closer to the sensor and ignores the semantic information.

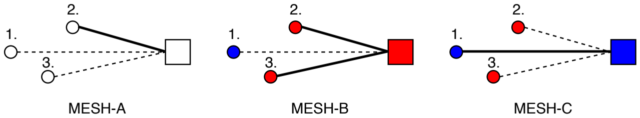


Fig. 6: Scheme for integrating semantic information into the TSDF with A: Ignoring semantic information, B: Integrating observations of the same classes, C: Integrating observations of the same classes with strong focus on first scan.

MESH-B (MB) uses a class-aware method to integrate the range images. Each source scan is sequentially integrated to the TSDF in the regular update fashion. In the integration, the class-aware TSDF checks for each new observation if the class matches the class of the existing observation. If they match, the procedure is the same as the regular update rule and the observations are integrated. But if the classes do not match, we choose either the existing or the new observation depending on the depth of the observation. Similar to the aforementioned z-buffer method, we use the closer observation to correctly represent occlusions.

MESH-C (MC) is also a class-aware fusion method that "focusses" on the primary scan and uses the additional scans only to fill missing data or integrate observations when the classes match. As we integrate the primary scan first, the TSDF treats any further frames as potential updates. If the current voxel is not occupied by any observation of the primary scan, we add the new observation to the corresponding voxel. In contrast, if the voxel is already set to a specific class by the primary scan, we let the new observation update the voxel only when the classes match.

We use an enhanced version of the marching cubes algorithm [16], to generate a smooth triangle mesh by ensuring topologically correct results. The original algorithm [17] often leads to cracks and ambiguities that are resolved by adding more cases to the lookup table. Finally, we use ray casting to sample a point cloud from the mesh. For this, we define the virtual laser scanner by defining the FOV, resolution, and relative position in space. We construct the array of rays by creating a grid spanning over the FOV – vertically and horizontally – and resolution-depending spacing. To speed up the ray casting, we use a modified implementation of the bounding volume hierarchy (BVH) [22].

The main reason for the meshing approach is to get surface representation instead of points to sample new scans from. Fusing multiple scans will improve the density of that sparse surface, e.g. between the individual LiDAR beams. Compared to the regular TSDF the MA method preserves the occlusions. However, the objects appear slightly larger due to noise from multiple scans. When using MB, the majority vote inside the TSDF cells leads to rougher boundaries between objects. Because of the focus for the first scan, the MC does not suffer from rough boundaries and also copes with the noise from multiple scans.

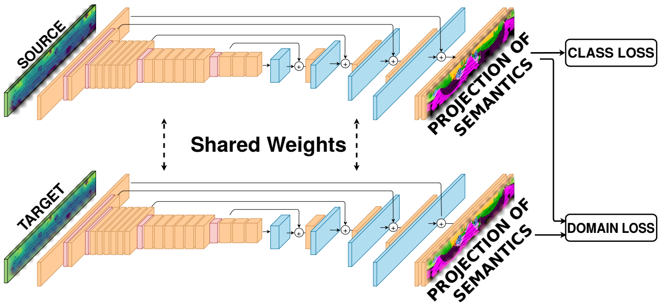


Fig. 7: Structure of the used CNN architecture [18] and the training with geodesic correlation alignment.

B. Geodesic Correlation Alignment (GCA)

The second step of the domain transfer is to retrain the model to adapt it to the target domain. While doing this and to prevent domain-shift between the semi-synthetic data rendered from the model and the real data coming from the new sensor configuration, we align the second-order statistics between source and target domains [35]. We apply this unsupervised domain adaptation during the training of the CNN model (see Fig. 4), using the rendered scans and labels. Let the labeled, semi-synthetic dataset generated by our first step be the source, and the unlabeled, real dataset the target. We extend our LiDAR-bonnetal framework [18] to take two inputs and use two losses, and generate equal sized batches from the source and the target dataset during training. We sample scans randomly from both datasets assuming the scenes will contain similar class content. We can use this naive sampling strategy because the domains are closely related and both scans show street scenes with a similar environment (city and highway driving). We sketch our training process in Fig. 7.

The semi-synthetic source batch is evaluated by the class loss, a weighted cross-entropy loss, and the semantic target. Besides, the training evaluates the target batch by using the geodesic loss by Morerio *et al.* [19], [35]. To limit the degradation in performance due to domain shift, they propose a method to align both distributions by minimizing the geodesic distance between the covariance matrices of target and source data. This leads to a loss function that can be used for domain adaptation in an end-to-end fashion in a single training step. To calculate the geodesic loss, we reorder and reshape the activations of the last feature layer to be of dimension c by $n \times h \times w$, with the number of channels c, the height h and width w of the layer and the batch size n.

Let $\mathbf{C}_S$ and $\mathbf{C}_T$ be the covariance matrices of the d-dimensional activations from a feature layer and $\|\cdot\|_F^2$ the squared Frobenius norm. To calculate the log of positive definite matrices, like covariance matrices, a common approach is to diagonalize it by a singular value decomposition and then calculate the logarithm of the eigenvalues. Whitening the covariances is encouraged to ensure full rank for the decomposition. $\mathbf{D}_S$ and $\mathbf{D}_T$ being the diagonalized source and target eigenvalues. The corresponding eigenvectors are $\mathbf{U}$ (source) for and $\mathbf{V}$ (target). Finally, the loss function is given by the geodesic Log-Euclidean distance between both covariance matrices

$$\mathcal{L}(\mathbf{C}_S, \mathbf{C}_T) = \frac{1}{4d^2}\left\|\mathbf{U}\log(\mathbf{D}_S)\mathbf{U}^T - V\log(\mathbf{D}_T)V^T\right\|_F^2 . \qquad (2)$$

Let $\mathbf{X}_S$ be the activations on the last layer and $\mathbf{Z}_S$ the source labels.

Finally, the training uses the minimal-entropy correlation alignment for unsupervised domain adaptation minimizing over the network weights θ

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\ H(\mathbf{X}_S, \mathbf{Z}_S) + \alpha\,\mathcal{L}(\mathbf{C}_S, \mathbf{C}_T), \qquad (3)$$

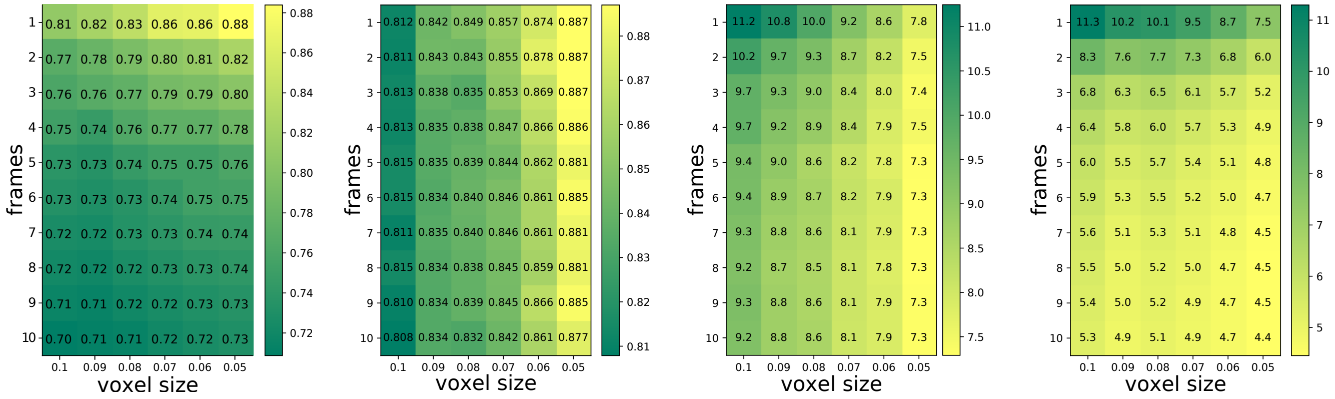
with the hyperparameter $\alpha > 0$ that minimizes the cross-entropy on the source domain $H(\mathbf{X}_S, \mathbf{Z}_S)$ and the geodesic loss $\mathcal{L}(\mathbf{C}_S, \mathbf{C}_T)$ of both domains. At the end of each training step, we penalize by the combined loss Eq. 3, which adds the domain loss weighted by α to the class loss.

IV. Experimental Evaluation

In this work, we present a method to adapt deep neural network models for semantic segmentation of LiDAR scans to a different scanner, using a real-to-real transfer. Our experiments are designed to show the capabilities of our method and to support our key claims, which are to be able to: (i) generate real-to-real LiDAR sweep transfer, including remission information, (ii) transfer the dense semantic annotations to the new sensor setup, and (iii) reduce the domain shift, when adapting a different LiDAR sensor.

Source Dataset. For the evaluation of our approach, we use the SemanticKITTI dataset [1] as our source dataset, which provides dense semantic segmentation annotations for the complete KITTI [8] odometry dataset. In total, the dataset consists of over 43 000 scans with point-wise annotations. Although only 23 201 scans are publicly available for training, and the remainder are used for evaluation on an evaluation server. This dataset covers 28 classes and distinguishes between moving and non-moving objects. The classes include traffic participants, but also functional classes for ground, like parking areas, sidewalks. The scans were captured with a Velodyne HDL-64 LiDAR scanner with a vertical FOV of $-25°$ down and $3°$ up at a capture rate of 10 Hz. We use 2048×64 px range images for the resulting projected range images used by RangeNet++.

Training setting. Throughout all these experiments, SemanticKITTI is our source dataset and we base our retraining on the RangeNet++ [18] model. For all approaches, we generate the datasets in advance and perform then our retraining with the generated scans. The overall data is split as proposed by Milioto *et al.* [18]. Thus, we use the 10 publicly available SemanticKITTI sequences, where sequence 08 is our validation set and the other sequences are used for training. The test data for the domain transfer is a single sequence of the nuScenes [3] dataset, which we manually labeled for this purpose with identical classes used by SemanticKITTI using the provided point labeling tool.



(a) mIoU: MESH-A/B (b) mIoU: MESH-C (c) MSE: MESH-A (d) MSE: MESH-B/C

Fig. 8: Grid searches of the MESH approaches for *voxel size* and *number of frames* simulating the same sensor.

Approach	frames	train/infer	mIoU	Acc	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic sign
Baseline	–	K/K	50.4	88.9	85.7	20.8	42.6	38.5	29.3	41.8	57.9	0.0	94.0	48.5	81.8	0.2	79.8	50.9	81.7	49.9	72.3	43.0	38.6
CP	1	K/S	54.4	91.1	94.0	23.7	46.6	44.8	31.1	47.5	65.7	0.0	94.6	49.4	83.7	0.0	86.6	55.8	85.0	55.8	75.6	53.7	40.3
MA	3	K/S	49.4	88.0	93.5	30.1	47.5	26.9	24.6	46.2	64.4	0.0	92.0	36.7	78.8	0.0	80.4	42.8	79.0	44.5	69.5	51.0	30.9
MB	3	K/S	50.7	88.3	93.8	27.5	49.0	35.5	25.4	42.0	64.9	0.0	91.7	40.6	76.2	0.1	83.1	46.2	81.3	49.2	68.4	49.8	39.2
MC	3	K/S	51.0	88.2	93.8	28.6	50.2	35.1	26.5	43.4	64.8	0.0	91.7	41.2	76.2	0.1	83.8	44.3	81.0	50.4	68.5	50.2	39.1

TABLE I: Pretrained baseline model on the synthetic datasets generated by CP and mesh approaches (MA, MB, MC with 0.05 m voxel size and three merged scans). *K*: SemanticKITTI, *S*: simulated dataset.

A. Baseline

The baseline for our quantitative results is the performance of RangeNet++ pre-trained on the SemanticKITTI dataset and evaluated on its validation data. RangeNet++ achieves 50.4 % mIoU as shown in Tab. I when staying within its domain. However, if we infer scans from a different domain (e.g. nuScenes scans) the performance drops to 12.3 % mIoU as shown in Tab. II.

B. Simulating Training Data

This experiment is designed to analyze the performance of different hyperparameters of our fusion approaches and evaluate how similar the generated scans are in comparison to the source scan when the source and target domain are the same. Thus, we run a grid search by varying the voxel size and the number of fused scans.

The goal of this experiment is to reproduce the same LiDAR scans and to compare them with the source scans. To measure the semantic similarity, we use the mean intersection-over-union (mIoU). The prediction X and corresponding ground truth Y represent the class as an integer value. For each class c, we extract the predictions X_c and the corresponding ground truth values Y_c, and compute the per-class IoU$_c$ as

$$IoU_c = \frac{X_c \cap Y_c}{X_c \cup Y_c} = \frac{TP}{TP + FP + FN}, \tag{4}$$

with TP being the true positive, FP the false positive and FN the false negative values of the matrix. The mIoU is given by the mean over the class-wise IoUs. To measure the geometric similarity between the two point clouds, we use the mean squared error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(I_i - \hat{I}_i)^2, \tag{5}$$

with the synthetic range image I and ground truth range image $\hat{I}$. We compute both by comparing the individual label and the range images of source and generated target for the mIoU and the MSE, respectively. Based on results shown in Fig. 8, reducing the voxel size improves details in generated range and label images. By adopting a class-wise TSDF, we achieve consistent performance in mIoU and improving MSE over integrating multiple frames.

Additionally, we use the pre-trained baseline model to evaluate the simulated scans. Therefore, we infer the different simulated datasets we created by the CP and MESH approaches. Tab. I shows the segmentation scores of the individual classes for the different approaches. The performance is similar to the baseline, which is inferred on the real scans.

C. Domain Transfer

The second experiment is to support the claim that our approach is able to transfer the original model to the target domain of a different sensor setup. The nuScenes [3] dataset is the target for our approach in Tab. II. In contrast to

Approach	α	frames	train/infer	mIoU	Acc	car	bicycle	other-vehicle	person	road	sidewalk	building	fence	vegetation	trunk	terrain	pole	traffic sign
Baseline	–	–	K/N	12.3	58.1	26.4	0.0	2.6	0.1	68.1	0.1	42.2	2.8	4.9	0.5	0.1	8.0	4.1
CP	–	1	S/N	**31.3**	83.7	64.1	1.6	3.3	13.1	85.6	34.0	81.8	41.4	39.8	3.4	0.9	26.5	11.6
MA	–	1	S/N	30.1	83.0	58.9	0.0	10.8	7.4	83.9	12.3	81.7	44.3	44.7	3.0	0.5	32.3	12.1
CP	–	5	S/N	28.8	81.4	58.9	0.3	5.8	13.8	84.9	30.6	79.3	33.9	28.3	3.3	0.9	24.6	9.8
MA	–	5	S/N	25.4	78.3	49.4	0.1	3.6	13.2	82.6	15.5	73.1	22.5	35.5	2.3	0.1	26.9	4.8
MB	–	5	S/N	**30.0**	82.0	56.7	0.1	10.6	14.1	85.8	26.4	77.2	35.8	32.8	2.7	2.1	33.7	11.6
MC	–	5	S/N	28.5	80.7	55.9	0.2	3.9	16.9	84.5	25.9	76.2	28.5	31.5	2.5	1.8	29.8	12.9
CP+GCA	1	1	S/N	**35.9**	85.7	70.8	8.8	6.4	18.6	88.5	53.0	80.9	43.0	41.6	3.4	0.8	35.3	16.3
MB+GCA	0.1	5	S/N	32.6	84.8	63.7	5.8	4.8	15.3	88.9	45.2	79.0	31.2	38.3	2.9	1.2	33.7	14.1
MC+GCA	0.1	5	S/N	**35.4**	85.9	65.2	6.7	3.1	20.9	89.3	49.8	80.0	42.1	44.7	3.9	2.9	34.2	17.3

TABLE II: Semantic segmentation performance on nuScenes data by the baseline, CP, and mesh approaches (MA, MB, MC uses voxel size of $0.07\ m$). *K*: SemanticKITTI, *N*: nuScenes, *S*: Simulated dataset.

the SemanticKITTI dataset, the LiDAR scanner features a lower resolution of 32 beams and a wider vertical FOV of $-30°$ down and $11°$ up at a capture rate of 20 Hz. We excluded several non-existing or very rare classes in this data for this evaluation. For this experiment, the baseline model is retrained with the simulated datasets generated by our different approaches. The base model's performance, without any adaption, drops to 12.3 % mIoU. With our CP approach with a single frame, we achieve 31.3 % mIoU and 28.8 % mIoU with multiple frames, providing the biggest jump in performance. We show that all mesh methods perform similarly, although MESH-A achieves the lowest mIoU with multiple frames. By adding the unsupervised domain adaptation we increase the mIoU performance of our CP+GCA approach with a single frame to 35.9 % and observe similar results with the MESH-C+GCA with multiple frames, providing a smaller, but a significant bump in IoU. For selected classes, we can recover the drop caused by the domain shift quite well.

D. Qualitative Results

In Fig. 9 we show the predictions of the best performing models of our approach. Two different scans from the labeled nuScenes sequence are colored by the predicted classes.

V. Conclusion

In this work, we presented a novel approach for the domain transfer of a semantic segmentation model for LiDAR data. Our approach operates by simulating a dataset for the transfer, but using real data, as opposed to the usually exploited simulation environments. Our methods exploit the fusion of multiple scans of the source dataset and meshing for a denser map to sample virtual scans from this. Multiple scans lead to noisier and more inconsistent maps, so this limits the number of frames to fuse. Especially challenging are semantics and remissions, which are not as consistent from different viewing angles as the depth. We cannot see a significant advantage of the different meshing methods. This needs further investigation in future work. The experiments

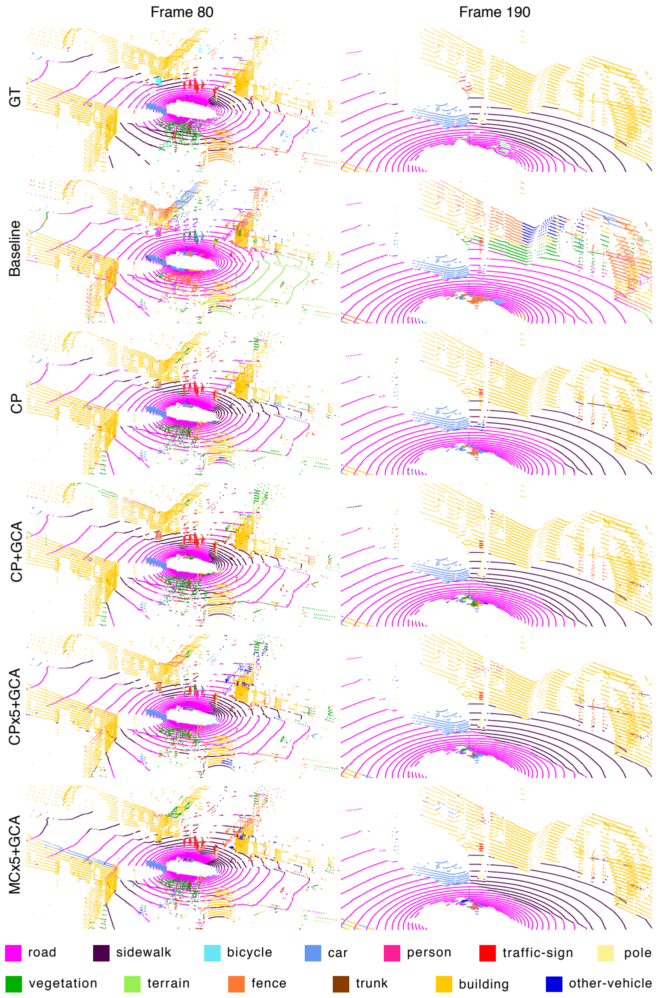


Fig. 9: Predictions of models from experiment *C*. From top to bottom: Ground truth (GT), Baseline, CP 1 scan, CP+GCA 1 scan, CP+GCA 5 scans, and MC+GCA 5 scans. Best viewed in color.

show that we successfully readjusted a model trained on the source dataset to the target dataset, which features a different LiDAR sensor with a different resolution, field of view, and location on the platform. Recently, the A2D2 dataset [9] made a large number of semantic annotations available, which would provide data to investigate the adaption of low-resolution to high-resolution LiDAR sensors in future work.

References

[1] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019.

[2] J. Behley and C. Stachniss. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In *Proc. of Robotics: Science and Systems (RSS)*, 2018.

[3] H. Caesar, V. Bankiti, A.H. Lang, S. Vora, V.E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuScenes: A Multimodal Dataset for Autonomous Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[4] X. Chen, A. Milioto, E. Palazzolo, P. Gigure, J. Behley, and C. Stachniss. SuMa++: Efficient LiDAR-based Semantic SLAM. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[5] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An Open Urban Driving Simulator. In *Proc. of the Conference on Robot Learning (CORL)*, 2017.

[6] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised Visual Domain Adaptation Using Subspace Alignment. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2013.

[7] L. Gan, R. Zhang, J.W. Grizzle, R.M. Eustice, and M. Ghaffari. Bayesian spatial kernel smoothing for scalable dense semantic mapping. *IEEE Robotics and Automation Letters (RA-L)*, 5(2):790–797, 2020.

[8] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012.

[9] J. Geyer, Y. Kassahun, M. Mahmudi, X. Ricou, R. Durgesh, A. S. Chung, L. Hauswald, V. Hoang Pham, M. Mühlegg, S. Dorn, T. Fernandez, M. Jänicke, S. Mirashi, C. Savani, M. Sturm, O. Vorobiov, M. Oelker, S. Garreis, and P. Schuberth. A2D2: Audi Autonomous Driving Dataset. 2020.

[10] J. Hoffman, E. Tzeng, T. Park, T.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell. CyCADA: Cycle-Consistent Adversarial Domain Adaptation. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2018.

[11] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham. RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[12] M. Jaritz, T. Vu, R. d. Charette, E. Wirbel, and P. Perez. xMUDA: Cross-Modal Unsupervised Domain Adaptation for 3D Semantic Segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[13] P. Jiang and S. Saripalli. LiDARNet: A Boundary-Aware Domain Adaptation Model for LidarPoint Cloud Semantic Segmentation. 2020.

[14] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen, and R. Vasudevan. Driving in the Matrix: Can virtual worlds replace human-generated annotations for real world tasks? In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2017.

[15] L. Landrieu and M. Simonovsky. Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[16] T. Lewiner, H. Lopes, A. W. Vieira, and G. Tavares. Efficient implementation of marching cubes cases with topological guarantees. *Journal of Graphics Tools*, 8(2):1–15, 2003.

[17] W.E. Lorensen and H.E. Cline. Marching Cubes: a High Resolution 3D Surface Construction Algorithm. In *Proc. of the Intl. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, volume 21, pages 163–169, 1987.

[18] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss. RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. In *Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[19] P. Morerio, J. Cavazza, and V. Murino. Minimal-Entropy Correlation Alignment for Unsupervised Deep Domain Adaptation. *CoRR*, abs/1711.10288, 2017.

[20] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-Time Dense Surface Mapping and Tracking. In *Proc. of the Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136, 2011.

[21] V.M. Patel, R. Gopalan, R. Li, and R. Chellappa. Visual domain adaptation: A survey of recent advances. *IEEE Signal Processing Magazine*, 32(3):53–69, 2015.

[22] M. Pharr, W. Jakob, and G. Humphreys. *Physically Based Rendering: From Theory To Implementation*. Morgan Kaufmann, 3 edition, 2016.

[23] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[24] C.R. Qi, K. Yi, H. Su, and L. J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2017.

[25] R.A. Rosu, P. Schütt, J. Quenzel, and S. Behnke. LatticeNet: Fast Point Cloud SegmentationUsing Permutohedral Lattices. In *Proc. of Robotics: Science and Systems (RSS)*, 2020.

[26] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from Simulated and Unsupervised Images through Adversarial Training. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[27] N. Silberman, D. Hoiem, P. Kohli, and Rob Fergus. Indoor Segmentation and Support Inference from RGBD Images. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2012.

[28] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M-H. Yang, and J. Kautz. SPLATNet: Sparse Lattice Networks for Point Cloud Processing. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[29] B. Sun, J. Feng, and K. Saenko. Correlation Alignment for Unsupervised Domain Adaptation. *CoRR*, abs/1612.01939, 2016.

[30] M. Tatarchenko, J. Park, V. Koltun, and Q-Y. Zhou. Tangent Convolutions for Dense Prediction in 3D. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[31] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese. SEGCloud: Semantic Segmentation of 3D Point Clouds. In *Proc. of the International Conference on 3D Vision (3DV)*, 2017.

[32] H. Thomas, C.R. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L.J. Guibas. KPConv: Flexible and Deformable Convolution for Point Clouds. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2019.

[33] A. Valada, R. Mohan, and W. Burgard. Self-supervised model adaptation for multimodal semantic segmentation. *Intl. Journal of Computer Vision (IJCV)*, July 2019. Special Issue: Deep Learning for Robotic Vision.

[34] V. Vineet, O. Miksik, M. Lidegaard, M. Niessner, S. Golodetz, V. Prisacariu, O. Kahler, D. Murray, S. Izadi, P. Perez, and P. Torr. Incremental Dense Semantic Stereo Fusion for Large-Scale Semantic Scene Reconstruction. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2015.

[35] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer. SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud. *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019.

[36] S. Yang, Y. Huang, and S. Scherer. Semantic 3D occupancy mapping through efficient high order CRFs. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2017.

[37] Li Yi, Boqing Gong, and Thomas Funkhouser. Complete & Label: A Domain Adaptation Approach to Semantic Segmentation of LiDAR Point Clouds. *arXiv preprint*, 2020.

[38] Y. Zhang, Z. Zhou, P. David, X. Yue, Z. Xi, B. Gong, and H. Foroosh. PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic Segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.